Theory and Practice of Using Models of Concurrency in Hardware Design

by Alexandre Yakovlev,
Professor of Computing Systems
Design

School of Electrical, Electronic and Computer Engineering
University of Newcastle upon Tyne, NE1 7RU

Thesis submitted for the degree of
Doctor of Science in Engineering

August 2005

Table of Contents

Abstract

This collection of publications presents the main research carried out by its author in the last twenty years (1985-present) in the area of modelling, analysis, synthesis and design of digital systems using formal models of concurrency. This research has been primarily focused on asynchronous or self-timed (without global clock) circuits and on Petri nets as the major model of their behaviour. The reason for using Petri nets and their interpretations has been mainly due to the fact that the behaviour of self-timed circuits is usually massively concurrent.

Self-timed circuits and the principles of their design will play an increasing role in future microelectronics systems, as predicted by the recent International Technology Roadmaps in Semiconductors (ITRS 2003 and 2005). The practical advantages of self-timing lie in saving power and reducing heat dissipation, achieving more robust synchronisation and allowing design reuse for complex heterogeneous systems, with hundreds of timing regions on a single chip. Additionally, self-timing offers ways of computing in hardware in a way that balances switching energy and electromagnetic emission, which is beneficial for high security and mixed signal (e.g. RF) applications. It has been for long time recognised that the main obstacle on the way towards wider exploitation of self-timed design principles in engineering practice is the absence of a practical and user-friendly methodology and tools for designing self-timed circuits, as their manual design is prohibitively complex due to the inherent high degree of concurrency in their behaviour. Their behaviour is governed by causal relationships and partial order of events rather than by the total order formed by clock pulses as in synchronous circuits. Petri nets naturally offer such an ability to cope with concurrency at the behavioural level because they capture causality and choice in their structure, thereby often reducing complexity of the analysis of the model to polynomial to the size of the net. Moreover, Petri nets have a solid theoretical foundation in terms of formal properties and semantics of concurrent systems, with a rich amount of knowledge and expertise accumulated in the last forty years. They appear to be an ideal candidate for an intermediate representation for specifications of asynchronous systems, between standard (front-end) hardware description languages and gate-level circuit representations. As such Petri nets may act as a formal semantic kernel for a new asynchronous design flow, and in this role be similar to the finite state machines being the core of synthesis methods for synchronous circuits.

The main result of this research is therefore the Petri net based methodology of designing asynchronous control circuits. This result could not have been achieved without performing investigations in the following closely related areas:

(1*) Formal Models of Asynchronous Behaviour*: Signal Transition Graphs, Causal Logic Nets, AND- and OR-causality, Models with Relative Timing, Relationship between Transition Systems and Petri Nets.

(2) *Asynchronous Circuit Design (including Design using Petri nets)*: Case studies involving designs of processors, bus and ring interfaces, counterflow pipeline, duplex communication channel, arbiters, asynchronous communication mechanisms.

(3) *Asynchronous Logic Synthesis, Design Flow and Tool Support*: Methods for complete state encoding, logic decomposition and technology mapping, direct translation of Petri nets to circuits, synthesis based on Petri nets unfoldings, visualisation and interactive synthesis, development of algorithms for synthesis, analysis and visualisation tools, conceptual models of design flows based on Petri nets and HDLs such as VHDL and Verilog.

(4) *Asynchronous Circuit Analysis and Verification*: Algorithms for Petri net and STG unfolding, Analysis of nets with read arcs (e.g. circuit Petri nets) using unfoldings, verification of circuits using unfoldings and combinations with symbolic traversals, Analysis of Performance of Asynchronous Circuits.

(5) *Asynchronous Communication Mechanisms (ACMs)*: developments of protocols and models for wait-free communication mechanisms for application in real-time computational networks and systems-on-chip, synthesis and hardware implementation of ACMs, evaluation and testing of ACMs, application of ACMs in building control systems.

(6) *Metastability, Synchronisers, Arbiters*: Models of metastable behaviour, analysis and design of synchronisers and arbiters, multiway and priority arbiters, Asynchronous A/D converters, Time measurement circuits, Time to code converters and time amplifiers

(7) *Asynchronous System Testing, Fault-Tolerance Design for Security*: self-diagnosis and self-repair of asynchronous circuits, structural fault-masking for asynchronous systems with request-acknowledgement interfaces, on-line testing of asynchronous control logic using Petri net specifications, energy-balancing for security and automatic insertion of security measures into the industrial design flow.

Statement of Originality and Contribution

Neither this material as a whole nor any part of it has been previously submitted for a degree in this or any other university. Although initial foundation for this research was laid while I worked towards my PhD, obtained in 1982 from St. Petersburg Electrotechnical University (formerly Leningrad Electrical Engineering Institute, also knows as LETI), all the listed work was written up and published after 1984, the year when I first arrived in Newcastle as a post-doctoral fellow through the British Council's exchange programme with the USSR Higher Education Ministry.

In 1985 I returned back to St. Petersburg, to continue my work as a lecturer gaining further experience from research contracts with industry (design of onboard computer systems for aircraft) and USSR Academy of Sciences (analytical instrumentation) and collaboration with fellow researchers working in the areas of asynchronous systems, design automation, fault-tolerance, parallel computation and artificial intelligence.

In 1991 I was appointed as a lecturer at Newcastle University, and carried out my research firstly and largely alone, and then to an increasingly greater extent in collaboration with my colleagues from Newcastle and abroad (particularly, Turin, Barcelona, Aizu (Japan), Intel (USA)). I had always believed that fruitful exchanges between colleagues with varied skills significantly improve the value of research in electronic system design. For example, without close collaboration with colleagues working in the area of synthesis of asynchronous circuits, it would not have been possible to produce a range of software tools for asynchronous design, particularly methods and software underlying the Petrify tool, which gained an award of a Finalist in the 2002 Descartes Prize competition in Europe.

While I can attest to having played a significant role in the papers listed, and have indicated my share of the work, I must stress that I fully recognise the roles played by my colleagues with whom I collaborated over nearly twenty years. For the majority of work listed (particularly in Group I), I was the catalyst, identifying the problem and guiding the investigation. The postdoctoral and postgraduate research assistants working directly with me were under my direct supervision throughout the study. Where the programmes included other staff members or colleagues from institutions abroad, the research assistants were jointly supervised.

Acknowledgements

My PhD thesis supervisor at St. Petersburg Electrotechnical University (SPETU) Victor Varshavsky introduced me to research in asynchronous systems and circuits in the early 80s. Leonid Rosenblum was my other teacher who introduced me to the world of concurrency models such as Petri nets. Vyacheslav Marakhovsky taught me how to design robust and efficient digital circuits without clock. Without initial start-up from these three people I would not have been in position to have carried out my own research in the subsequent twenty years as reflected in the publications presented in this thesis.

I am indebted to David Kinniment and Brian Randell for the many ideas about circuits and systems design that I acquired during my stay in Newcastle as a post doc in 1984-85 and later in the 90s when I became a lecturer at the CS department. Our coffee-time discussions with David Kinniment about synchronization, arbiters, AD converters and many other subjects till present day give me enormous motivation and drive.

Albert Koelmans, Maciej Koutny and Gordon Russell are the great friends and colleagues who I have kept talking to about computer architecture, hardware description languages, concurrency, testing and fault-tolerance. It was Gordon who about two years ago suggested to me that I should go for a DSc.

I feel honoured to be associated with the following remarkable scientists, who have been colleagues and friends to me for years: J. Cortadella, A. Davies, M. Kishinevsky, A. Kondratyev, L. Lavagno, Yu. Mamrukov, I. Mitrani, O. Maevsky, A. Petrov, N. Starodoubtsev, Yu. Tatarinov, I. Yatsenko, A.Taubin, W. Vogler, and others.

I am very thankful to the young researchers, lecturers and graduate students I have worked with at Newcastle, F. Burns, A. Bystrov, I. Clark, V. Khomenko, L. Lloyd, A. Madalinski, A. Semenov, D. Shang, D. Sokolov, F. Xia, and many of my present research students, without whom this work would have not seen further progress.

My friends and colleagues at Computing Science and EECE schools at Newcastle have always been sources of education, inspiration and motivation for me.

Friends and colleagues from two academic communities, Asynchronous Systems Design and Petri Nets, as well as industrial collaborators from Intel, Atmel and MBDA have always been so helpful in providing motivation and critical feedback, as well as sharing problems, ideas and experiences.

Lastly but, perhaps, most importantly, I am immensely grateful to all my family. My father Vladimir Yakovlev, now Honorary Professor of Automation and Control at SPETU, has always been an example of a dedicated academic to me. When I was a little boy my father often took me to his Control Systems Labs in the SPETU courtyard on the Prof. Popov Street, where he worked with his research assistants on multi-channel controllers and non-linear pulse-sampled systems, and where I saw many interesting experimental devices and instruments. Throughout my career he has always had time to

offer me a good advice. My mother Irina, who taught electrical engineering in a technical college, introduced me into the magic of electromagnetism when I was about ten. My wife Maria, who also graduated from SPETU, and my son Greg, currently a research student in biochemistry at Cambridge, have always been patient with my long working hours and provided me with constant moral support and stability, so much needed for any intellectually challenging work.

Finally, most of my team's research at Newcastle has been generously supported by EPSRC grants. The list of projects, which I have always been keen to be given an easy-to-remember acronym, includes:
ASAP (GR/J52327) "Automated synthesis of parallel synchronous and asynchronous controllers", in collaboration with Bristol University, (1994-97), "Automated synthesis of asynchronous control circuits" Visiting Fellowships for L.Lavagno and M. Kishinevsky (GR/J72486 and GR/J78334, 1994-96), HADES (GR/K70175) "Hazard-free arbiter design"(1996-99), ASTI (GR/L24038) "Asynchronous circuit synthesis and testing", Visiting Fellowships for A. Kondratyev and L. Lavagno (1997-99), TIMBRE (GR/L28098) "Time-predictable hardware platforms"(1997-00), COMFORT (GR/L93775) "Asynchronous communication mechanisms for real-time systems", in collaboration with Kings College London,(1997-2001), MOVIE (GR94366) "Model visualisation for asynchronous circuit design" (2000-02), BREACH (M94359) "Behavioural refinements for asynchronous circuit synthesis" (2000-02), COHERENT (GR/R32666) "Computational Heterogeneously Timed Networks", in collaboration with Kingston University, (2001-2004), BESST (GR/R16754) "Behavioural Synthesis of Systems with Heterogeneous Timing" (2001-2004), STELLA (GR/S12036) "Synthesis and Testing of Low-Latency Asynchronous Circuits" (2002-2006), and SCREEN (GR/S81421) "Secure Circuit Design" (2004-07).

Other helpful research support came from British Council (particularly for supporting collaboration with J. Cortadella at UPC, Spain), Royal Society, ESPRIT (particularly from ACiD-WG Working Group on Asynchronous Circuit Design), Nuffield Foundation, Leverhulme Trust, Acorn Networks and Newcastle University Research Committee.

Summary of Contributions in Submitted Portfolio

Introduction

Research on aperiodic automata, self-timed systems and implementation of concurrent processes in hardware was pioneered in Russia by the late Professor Victor Varshavsky, who was the author's PhD advisor. Throughout the late 70s, 80s and early 90s Varshavsky led a research group at the Department of Computer Software at St Petersburg Electrotechnical University (formerly known as Leningrad Electrical Engineering Institute named after V.I. Ulyanov (Lenin)). This work provided a clear understanding of the principles of asynchronous communication and computation and the ways of their implementation in digital VLSI hardware.
The key idea behind these principles was based on concurrency and "soft timing" inherent in self-timed systems due to the use of local handshaking and signal acknowledgement. Handshakes helped computing engines to be liberated from the tyranny of global clocking and rigid time-stepping. The practical advantages of self-timing are in saving power and reducing heat dissipation, achieving more robust synchronisation and allowing design reuse for complex heterogeneous systems, and in offering the possibility to build systems with self-checking and self-recovery.
The results of V. Varshavsky's group's research laid a solid scientific foundation to a wide range of developments in microelectronics and computer engineering that took place in the 90's in the international arena. Industrial self-timed design examples from Philips, Intel, Sun Microsystems and other companies, including several start-ups working mainly in the area of asynchronous design, are rapidly mushrooming around the world in signal processing, mobile computing and embedded systems in general. They exploit those principles and circuit solutions, in the form of fully working asynchronous microprocessors and microcontrollers.
As semiconductor technology marches through the new era of Systems and Networks on Chips and faces the thrilling uncertainty of nanotechnology and quantum computing, the role and the future of systems with "soft" timing only seems brighter.

This thesis focuses on the research of the last twenty years and scientific results achieved by its author in using models of concurrency for designing self-timed systems.

Perhaps, the most notable of all these investigations has been the work on developing formal models of concurrent and asynchronous systems behaviour and a Petri net based methodology for designing asynchronous control circuits.
This research, lying on the border between concurrent systems theory and digital circuit design, is sufficiently mature today and some of the achievements outlined in the following sections have reached the level of a monograph, many journal and conference papers, lecture notes and tutorials.

1. Formal Models of Asynchronous Behaviour

The purpose of formal modelling as seen in this research is at least twofold. Firstly, formal modelling is crucial for understanding the behaviour of asynchronous systems and circuits, without which it would not be possible to argue about their design, analysis and synthesis. Work prior to the investigations presented in this thesis focused mostly on developing representations for asynchronous systems within the finite state machine (predominantly Huffman's) model. This approach lacked adequate notion of causality and concurrency, paradigms absolutely essential in reasoning about large systems without global clocks. Secondly, models provide foundation for developing methods, algorithms and, most importantly for practical application, software tools for the design automation of VLSI systems. The work on formal models produced new results about relationships between causal and state-based models, between speed-independence and delay-insensitivity, unified Petri nets and Change Diagrams, two main causal (event-oriented) models of asynchronous control circuits. This paved the way to work on synthesis and verification using causal models (Sections 3 and 4), leading to much more efficient algorithms than previously used state-based techniques. This research indicated the ways of modelling circuits with timing constraints avoiding explicit notions of timing regions, thereby escaping from another source of computational complexity in design.

1.1 Signal Transition Graphs

The key role in this research belongs to the model of *Signal Graphs*, which was introduced by Leonid Rosenblum and the author of the thesis in [1]. This model is better known today as *Signal Transition Graphs* or STGs (independently, a similar model was proposed at MIT by T.A. Chu in 1985). STGS are based on Petri nets whose transitions are interpreted as the rising and falling edges of binary signals. The model was investigated in more detail later in [2,3,23,24], which showed the restrictions and limitations of Chu's model and presented a unified STG model, with proofs of its formal relationship with Labelled Transition Systems and lattices defined on Parikh vectors of transition firings. As a by product it has unified the links between Muller's theory of speed-independent and semi-modular circuits with Udding's notion of delay-insensitivity. This model has found wide-spread use and led to further investigations by many researchers and asynchronous designers around the world in the last fifteen years (monographs and papers in IEEE journals, and conferences such as ASYNC, ACSD, ICCD, ICCAD, DAC, EDAC, EDTC, DATE).

1.2 OR-causality and Causal Logic Nets

The *concept of causality* is a fundamental one in modelling asynchronous hardware behaviour. Particularly innovative has been the investigation of *OR-causality*, a concurrency paradigm implemented within the new Petri net extension called *Causal Logic Nets* [3]. Novel concepts of joint and disjoint OR-causality have been defined in this work. This work also studied the formal link between STGs and Change Diagrams,

proving that Change Diagrams are *not a subclass* of labelled Petri nets under observational equivalence. Reductions of the CLNs to Petri nets, Change diagrams, Inhibitor nets have been proven. The original "binary version" of the STG model has been extended to multi-valued or symbolic STGs [9]. This work has resulted in wider research in this area in the last few years (e.g., at University of Kaiserslautern). Work on OR causality has led to more recent development of the idea of *early propagation* or lenient evaluation in asynchronous logic circuits which has significant impact on performance of pipelines (work at University of Manchester and Carnegie-Melon University). Also in our recent work (2003-05) on secure hardware design at Newcastle understanding of OR-causality helps to solve the problems of information leakage due to early propagation in cryptographic hardware.

1.3 Models with Relative Timing

The original STG model [1] was defined for *both untimed and timed* cases, thus not only enabling the design of circuits with unbounded delays (pessimistic, speed-independent, case) but also circuits with timing constraints, allowing optimisation for speed and area at the cost of being more definitive about delays.
The latter aspect has also led to a variety of investigations in the asynchronous community, including the concepts of *"Lazy" Transition Systems* and *Relative Timing* [42] that was actively used at Intel in developing instruction decoder RAPPID for Pentium 4.

2. Asynchronous Circuit Design: Methods, Case Studies

In order to prove the usefulness of formal methods, techniques and tools in practice, a number of *asynchronous control circuits* have been designed using Petri nets, STGs and related techniques. These design case studies include: interface logic and bus controllers [23,28,29,30,31,34], asynchronous pipeline token-ring interface [4,7], arbiters [18,19,60], A/D converters [62,63,64], micropipeline circuits and processors (e.g., Sproull's Counterflow pipeline) [4,5,32,33,35], ESPRIT ACID-WG industrial design problems, e.g. loadable mod-N up/down counter and interrupt controller (cf. Materials of ACiD-WG workshop in Groningen, which can be obtained from the author).

2.1 Use of theory of regions in design

These designs have showed the power of formal techniques based on Petri nets. For example, in designing Counterflow pipeline controller [5], the crucial role belonged to the *use of theory of regions* [25], because regions are a way to decompose global state into local state and thereby extracting natural concurrency from the specification.

2.2 Design of interfaces using Protocol Machine

The research in designing control logic using Petri nets has also advanced the protocol-driven approach to designing interface controllers. The main idea of this approach is based on the use of a "*protocol machine*" (PM) as an initial specification of a protocol in systems built following the concept of communication-centric design. The PM is a 'hypothetical' automaton (possibly with concurrent actions) which is placed between the real communication entities, such as master(s) and slave(s), to constrain the possible actions of all entities in their communication. The PM technique has been first applied (manually, but in the future we consider its automation!) in designing controllers for the duplex communication channel [6]. This work also contributed to *low latency* design methods (see Section 3 about direct mapping techniques) is achieved in combining direct mapping of the control logic from the Petri net model and organising the push and pull handshakes with the data path in such a way that the control actions are maximally out of the critical path of the data channel [6].

2.3 Demonstrator Chips

A demonstrator chip, called HADIC, was designed at Newcastle in 1999-2000 and fabricated by EUROPRACTICE. The chip included samples of arbitration and asynchronous communication circuits. The HADIC chip was successfully tested and used in recent experiments [65,73]. More recently the author supervised design of other demonstrator chips in the area of secure circuits, synchronisers (see Sections 6 and 7)

3. Asynchronous Circuit Synthesis

3.1 Petri net based synthesis methodology

The *Petri net based design methodology* plays a key role in the synthesis of asynchronous control circuits [4,11,34]. The methodology has two stages. The first stage, *Abstract Synthesis*, uses labelled Petri nets and their composition. The second stage, *Logic Synthesis*, uses the refinement of the nets obtained by Abstract Synthesis into Signal Transition Graphs (STGs) and synthesis of hazard-free logic circuits from STGs.

3.2 Synthesis methods and algorithms

The last fifteen years have led to the development of a set of new methods and algorithms supporting the above-mentioned methodology, namely:

- methods for synthesis of speed-independent circuits *directly* from STGs, i.e. *avoiding* the full state space exploration, using lock (coupledness)

classes [31] (originally proposed in the author's PhD thesis in 1982) and using Petri net unfoldings and approximate boolean covers [10,43];

- method for synthesis of safe Petri nets with read arcs from transition systems and a method of solving the *state encoding problem* [11] in STG-based synthesis, both based on *theory of regions in transition systems* [5,25,58];
- method for the *hazard-free implementation* of speed-independent circuits, using *simple gates* [8] and *monotonic cover conditions* [39];
- method for *decomposition and technology mapping* of speed-independent circuits, using *Boolean factorisation and binary relations* [38,41];
- methods for *asynchronous circuit optimisation* (for speed and area factors), and constructing *locally speed-independent (or with bounded delays) and globally delay-insensitive* circuits, using various *STG transformation* techniques (under appropriate equivalence criteria) concurrency reduction and expansion, handshake expansion (see H. Saito, A. Kondratyev, J. Cortadella, L. Lavagno, T. Nanya and A. Yakovlev, Design of asynchronous controllers with delay insensitive interface, IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences, Vol. E85-A(12): 2577-2585, December 2002);
- method for circuit decomposition for implementation in *negative (standard logic library) gates* [47];
- method for synthesis of control logic from STGs using *direct mapping* based on separating control and interface logic [46,48,49];

3.3 Synthesis from Hardware Description Languages

To gain greater practicality in the automated synthesis of asynchronous circuits, and achieve their wider adoption, the concept of asynchronous design flow, based on the Hardware Description Language (HDL) front-end and use of Petri nets and STGs as *intermediate* (internal for the design tools) language has been developed [12]. The method uses labelled Petri nets for control logic and coloured Petri nets for data path. This approach has the advantage of being oriented on a non-expert (standard) designer [26]. The method also uses *direct mapping of Petri nets* to asynchronous control circuits, helping to achieve productivity and optimality of asynchronous designs. It was first presented in [4] and later advanced in [48,50].

3.4 Asynchronous behaviour visualisation and interactive synthesis

To assist wider use of asynchronous design, techniques and algorithms for the visualisation of asynchronous and concurrent behaviour have been developed. Although it has its independent value from the theoretical point of view (new forms of representing concurrency semantics in its partial order form), this research comes close with the above-mentioned work on automated synthesis and verification of asynchronous circuits, especially where such tasks are interactive and involve human designer. For that, the idea of *separating concurrency and choice* has been developed

and implemented in software (A. Bystrov, M. Koutny and A. Yakovlev. Visualization of partial order models in VLSI design flow, Proc. DATE'02, Paris, March 2002, IEEE CS Press, pp. 1089-1090). Subsequently, comprehensive methods for the visualisation of asynchronous circuit behaviour and resolution of state coding conflicts using conflict cores in the unfolding prefix have been developed [45]. These methods support the idea of interactive synthesis of low-latency control logic.

3.5 Synthesis tools

Many algorithms supporting this methodology have been implemented in software tools, particularly in **Petrify** [11,37], developed by Jordi Cortadella at the Polytechnic University of Catalonia. Most of the circuits mentioned in Section 2 have been designed using Petrify. Other tools, such as PUNT, PN2PD, Verisyn, OptiMist, ConRes, developed at Newcastle under the author's direct supervision.

Monograph [11] presents the main synthesis flow from STGs. Furthermore, a large community of asynchronous system designers in the UK and abroad (e.g., the designers of the first industrial-strength asynchronous microprocessor Amulet at the University of Manchester, designers at Intel, Philips, Theseus Logic, AT&T to name but a few) are using STGs and Petrify for synthesis and analysis of their circuits.

4. Asynchronous Circuit Verification and Analysis

The first ideas for relation-based (not involving explicit state exploration) techniques for verification of concurrency models of asynchronous circuits were outlined in [13]. Further developments in the area of asynchronous circuit verification were based on the exploitation of *partial order techniques*, such as Petri net unfoldings and combinations of unfoldings with symbolic traversals [32,51].

New methods and algorithms for Petri net unfolding have been developed to improve the efficiency of the partial order analysis approach for k-bounded Petri nets and Petri nets with read arcs, using the techniques based on a *(FIFO or LIFO) ordering of tokens* in nonsafe places, *representative sets* of transitions [51], weak causality and contextual cycles [14].

These methods have been implemented in the above-mentioned PUNT tool. These techniques and tools have been successfuly used in a number of applications such as verifying control logic for Amulet microprocessors and checking coherence of a four-slot asynchronous communication mechanism (cf. [55]).

More recent work has applied *unfoldings and integer programming* and *SAT-solvers* to complete state coding verification [43,44].

This research has also produced a number of practically useful asynchronous circuit analysis methods:

- method for *estimating power consumption* in asynchronous control circuits *based on Petri net T-invariants* (L. Lloyd, A. V. Yakovlev, E. Pastor, A.M. Koelmans. Estimations of power consumption in asynchronous logic as derived from Graph Based Circuit Representations. International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS'98), Technical University of Denmark, October 7-9, 1998, pp. 367-376, A.M. Trullemans-Anckaert, J. Sparsoe (eds).);
- method for *performance analysis of asynchronous arbiters* [54];
- method for *estimating the worst-case execution time* for CPU models using coloured nets [15,52]
- method for verifying design abstractions for concurrent specifications in Ada [53].

5. Heterogeneous systems and asynchronous communications

To help solve design problems in the current decade with Systems-on-Chip, which will have billions of transistors on a single die and thousands of timing domains, the idea of *heterogenously timed networks (hets)* has been proposed. The key components of a het are *asynchronous communication mechanisms* (ACMs), which act as buffers between potentially timing-independent *motive powers,* provided by computational blocks. This work has been carried out in collaboration with MBDA, a leading European company in real-time systems for missile control. A new taxonomy of ACMs has been proposed which involves *automated synthesis of ACM protocols*, and their hardware and software implementation [17,55,58]. The idea of exploiting *wait-free communication at the hardware level* offers a very promising advantage of creating harmony between the traditionally conflicting requirements of real-time and low-energy consumption. This may revolutionise the area of embedded systems design, with unlimited opportunities for miniaturisation and flexible operation [16]. Some applications of hets and ACMs to designing control systems have been investigated [56,57]. A 3-slot pool ACM was implemented on the above-mentioned HADIC chip.

6. Metastability, Synchronisers, Arbiters, A/D Converters and Time Measurement Hardware

Another way in the direction of SoCs has been investigated through studying metastability, synchronisation and arbitration, being the fundamental problems in digital design of systems with asynchronous interfaces. Without rigorous modelling and analysis of *metastability*, the full understanding of how *synchronisers and arbiters* operate and how they should be designed would not be possible. A closely related problem solved within this domain was the design of *asynchronous A/D converters and time to digital converters*.

This research was done in close collaboration with Professor David Kinniment, a pioneer and international authority in this field. SoCs of the future will have thousands of timing regions possibly controlled by local (free-running or stretchable) clocks. They will need reliable and fast synchronisers. Fully asynchronous circuits, operating without clock at all, also require time coordination circuits, called arbiters. The main contributions of the author were in the area of arbiter designs, whereas in studying the performance of synchronisers he was more in supporting role to D. Kinniment.

6.1 Metastability

Compact *Petri net models of metastability* in D-latches and transparent latches have been developed in [61]. The problem of metastability in A/D conversion has been studied in [62]. A long-standing problem of the existence of *oscillatory anomaly in a three-way arbiter* using a tri-flop made of CMOS NAND gates, as opposed to an interconnection of two-way mutual exclusion elements, has been solved in [69], where the *conditions for oscillations* have been analytically (using small-signal models) derived.

6.2 Synchronisers

The problem of deriving the *time constant characteristic of a synchroniser*, a key parameter which determines mean time between failures in systems with asynchronous signals and clocks, has been solved using the combination of analytical and simulation techniques in [65]. This work is now being progressed towards physical characterisation of synchronisers.
Synchronisation between independently clocked regions in a high performance system is often subject to latencies of more than one clock cycle. It has been shown in [68] how the *latency can be reduced* significantly, typically to half the number of clock cycles required for high reliability, by *speculating* that a long synchronization time is not required.

6.3 Arbiters

One of the fundamental problems in designing asynchronous circuits has been whether it is possible to construct control circuits for STG specifications which are *non-output persistent* (i.e. with intentional conflict resolution). Finding a solution to this problem is important for automating the design of asynchronous arbiters. The key initial step in formalising this problem and a method (which is now only partially automated) for its solution was developed in [59]. It uses the idea of *factorisation of arbiters* from the STG for logic synthesis. This method has been used in many design examples listed in Section 2, such as Counterflow pipeline, token-ring arbiters, duplex communication channel.

Techniques such as disjoint OR-causality (see Section 1) and partial pipelining have been used in developing *low latency arbiters using tree-structures* [18]. Further advancement of early propagation methods in designing *priority arbiters* has been done in [19]. These priority arbiters have later been used by a number of researchers (e.g. Manchester University, Technical University of Denmark) developing routers for Networks on Chip. The idea of a multi-way arbiter with quick response time and in-order service of requests has been implemented in *ordered arbiters* [60]. Ordered and priority arbiters were implemented in our HADIC chip (see section 2).

6.4 Analogue to Digital and Time to Digital Converters (Time Measurement)

A number of asynchronous A/D converters, following successive approximation and flash approaches, in combination with bundled data, fundamental mode and fully self-timed designs, have been presented in [62,63,64]. These designs enjoyed low power and low noise characteristics compared to their synchronous prototypes. One A/D converter was implemented on the above-mentioned HADIC chip.

Research on *time measurement at the pico-second level* was originally triggered by a problem of testing set-up and hold conditions on chip, suggested to our group by a contact person at Cypress Semiconductors. This research has led to developing fundamentally new structures for on-chip time measurement based on *time-to-digital conversion and time amplification using mutex elements*. The former was solved in a successive approximation (using a log-size stack of mutexes) way in [20,66]. The originality of the time difference amplifier [67] was in that it used metastability as an "ally" rather than "enemy" of the designer, because the metastability resolution time was proportional to the inverse of the logarithm of the input time difference, thereby resulting in the amplification gains of up to 5. Such as scheme allowed measuring input time differences of say 2 ps, which could be amplified by *time difference amplifier* to 10ps, which could then be measured by a time-to-digital converter.

A chip with a time to digital converter circuit and time amplifier was fabricated at Sun Microsystems in 2003, and has been tested demonstrating the feasibility of our revolutionary techniques.

7. Asynchronous System Fault-tolerance, Testing and Design for Security

7.1 Fault-tolerance

One of the difficult obstacles on the way to the exploitation of asynchronous design is the problem of their efficient testing. This involves a number of issues such as studying self-checking properties of asynchronous circuits, developing self-test, self-diagnosis

and self-repair features. This research has produced several original methods for introducing *fault-tolerance and self-repair* in system architectures. While still working in Varshavsky's group the author was one of the key designers of a *fault-tolerant token ring channel* developed for an onboard multiprocessor system [7,70,71,72]. An original protocol was developed for the self-timed ring channel, which used *3-of-6 delay-insensitive code, distributed priority-based arbitration technique, relative address-based routing (for high speed) and pipelined data transmission, FIFO buffers and async-sync interface* to the synchronous bus architecture of computational part. In many ways, the system was the first example of a *globally asynchronous and locally synchronous* (GALS) system, with communication features (M-of-N encoding and routing based on relative addresses) similar to those used in today's *Network-on-Chip* (NoC) projects. While being self-timed in the normal operation mode, the channel switched into synchronous mode to perform fault-location and recovery by using redundancy in wires and transducers. A novel technique based on *sliding redundancy* was also developed. Overall, the channel allowed to tolerate (detect, locate and self-recover) up to two stuck at faults in communication links or in asynchronous parts, without involving any higher level facilities.

Another important contribution in the fault-tolerance domain was the principle of structural fault-masking in asynchronous interfaces [21]. The method was based on inserting a *mirror protocol converter* into each handshake interface which would adjudicate the responses coming from communication channel and mask those that do not match the desired behaviour produced by the mirror model.
This method can, at a very low cost, *mask transient faults and single event upsets on handshake interfaces*, which is becoming increasingly important in the future SoCs and NoCs.

7.2 Asynchronous Circuit Testing

Novel methods for *on-line testing of asynchronous circuits* have been developed, the area where at the outset there had been very little known from prior research. The investigation has built on the ideas of structural fault-masking for handshakes [21]. It was concerned with development of *on-line monitors (snoopers) and fault checkers* for handshake protocols specified by Petri nets. In its simpler form the checker was able to detect violations of 4-phase protocols (cf. *refusal sets* known from theory of concurrency semantics) and signalling them to a special error handling infrastructure. Different strategies and mechanisms were developed for the latter using early propagation and strong indication signalling methods [77]. In addition to detecting order violations timing errors, violating reasonable delay bounds (min and max) in handshake phases, have been made detectable.

A novel method for testing ACMs (See Section 5) for their specific properties of data coherence and freshness, has been designed and applied to the Pool ACM fabricated on the above-mentioned HADIC chip [73].

7.3 Secure Circuit Design and Power-Balancing Techniques

To help solve information leakage problems at the hardware level, and at the same time maintain the design flow accepted by design houses developing smart cards and other devices with cryptographic solutions, the author and his team have come up with the idea of using *dual-spacer monotonic transition protocol* introduced in dual-rail circuit designs. The class of circuits based on two spacers 00 and 11 is called *phase difference based logic*. It has a unique property of *invariance of its switching activity from the processed data*. This property opens up possibilities not only for security but for efficient testing because the time for testing for a large class of faults (this question depends on the self-checking properties of the underlying implementation) again becomes independent of the processed data domain. The design of secure circuits is carried out *completely automatically for clocked solutions using standard RTL synthesis* tools and additional tools have been developed in Newcastle [22,75]. This work has involved collaboration with Atmel Smart Card ICs, who used our design flow methodology to experiment with the company's designs. An asynchronous AES block has been designed which used *novel secure power-balanced latches* and partially-speed-indepenent (for saving area and power) data path pipeline [74]. Recently, our own experimental VLSI chip has been designed, taped out and fabricated (via Europractice) with a AES cryptographic core in order to investigate the impact of the new design methods for security. Self-checking properties were investigated in the context of security applications [76].

8. Future work

We are still far from the widespread use of asynchronous design in microelectronic industry. Here, the situation is like in setting up a non-linear switching process. The real commercial world cannot take the revolutionary ideas about building systems without clock onboard instantly. The process requires some catalysis in order to overcome the natural inertia. Therefore the *short-term* research needs are as follows:

- Easy-to-use CAD tools for constructing self-timed circuits by non-expert designers trained in traditional synchronous design. These tools are first supposed to provide a seamless evolutionary way from synchronous designs, and should rely on the use of the existing commercial design flow, including placement and routing software. For example, more work is urgently needed to automate the synthesis of self-timed data path. A key issue here is a trade-off between tolerance to variability (which calls for techniques such as dual-rail) and power and area costs.
- Methods for testing asynchronous circuits using existing automatic testing equipment. Testing asynchronous circuits, whose behaviour is inherently concurrent and whose controllability and observability with respect to primary inputs/outputs is limited, is a big problem. Self-testing, online testing and built-in testing approaches need to be investigated further. At the same time, adding self-test facilities must not impair the performance of high-speed logic in the normal operation mode.

- Interfaces between synchronous and asynchronous circuit domains, or the so-called globally asynchronous and locally synchronous (GALS) systems. Again, this would help a more gradual transition from synchronous design approach and would allow reuse of the massive amount of "synchronous" intellectual property within the new asynchronous System-on-Chip context. In the future designs will likely be timing plastic, i.e. with some design-time and run-time configurability of timing modes.
- More demonstrator designs and products with asynchronous ircuits, to prove their advantages in terms of power savings, EMC, modularity, design efficiency, and robustness. The best areas of demonstrating the advantages are likely to be systems with heterogeneous timing such as those from signal and image processing applications and systems involving high-bandwidth on-chip networking

In the *longer term*, a more fundamental research on truly asynchronous and concurrent systems needs to be pursued. This research should effectively develop a mature understanding of the Token-Based Computing in various implementation technologies. It may include (but not limited by):

- Synthesis of concurrent specifications of asynchronous designs from partial order and sequential fragments.
- Verification of complex asynchronous behaviour, such as that produced by a mix of data path and control flow with a range of causality paradigms.
- Methods for the direct mapping of concurrent specifications onto various implementation technologies, such as CMOS, nanotubes, quantum dots etc.
- Methods for analysis and synthesis of circuits with analogue components, whose behaviour is that of a complex dynamic non-linear system.

Portfolio of works submitted for Doctor of Science in Engineering

Group I. Requirement “Those works upon which you primarily base your claim to have satisfied the standards for the award of the degree which are indicated in section 2 of the regulations.”

Group II. Requirement “If applicable, other works or lists of works put forward as additional evidence of the scope of your contribution to the field or fields of study in which the primary submissions lie.”

[P] indicates postdoctoral or postgraduate research assistant coauthor
[C] indicates academic colleague co-author
[E] indicates external or international collaborator co-author

This thesis contains hardcopies of the publications of Group I.
Publications from Group II can be obtained from the author upon request.

Group I (Primary contributions):

(1) Models of Asynchronous Systems based on Petri nets:

1. L.Ya. Rosenblum [C] and A.V. Yakovlev. Signal Graphs: from Self-timed to Timed ones, Proc. Int. Workshop on Timed Petri nets, Torino, Italy, July 1985, IEEE CS Press, pp. 199-207.
2. A. Yakovlev, L. Lavagno [E] and A. Sangiovanni-Vincentelli [E]. A unified signal transition graph model for asynchronous control circuit synthesis. Formal Methods in System Design (Kluwer), Vol. 9, No. 3, Nov. 1996, pp. 139-188.
3. A. Yakovlev, M. Kishinevsky [E], A. Kondratyev [E], L. Lavagno [E] and M. Pietkiewicz-Koutny [P]. On the Models for Asynchronous Circuit Behaviour with OR Causality. Formal Methods in Systems Design (Kluwer), Vol. 9, No. 3, Nov. 1996, pp. 189-234.

(2) Asynchronous Circuit Design (using Petri net based models):

4. A.V. Yakovlev and A.M. Koelmans [C]. Petri nets and Digital Hardware Design Lectures on Petri Nets II: Applications. Advances in Petri Nets, Lecture Notes in Computer Science, vol. 1492, Springer-Verlag, 1998, pp. 154-236.
5. A. Yakovlev. Designing Control Logic for Counterflow Pipeline Processor Using Petri nets, Formal Methods in Systems Design (Kluwer), Vol. 12, No.1 (January 1998), pp. 39-71.
6. A. Yakovlev, S. Furber [E], R. Krenz [P], A. Bystrov [P]. Design and Analysis of a Self-timed Duplex Communication System, IEEE Transactions on Computers, Vol. 53, No.7, pp. 798-814, July 2004.

7. A. Yakovlev, V. Varshavsky [E], V. Marakhovsky [E] and A. Semenov [P], Designing an asynchronous pipeline token ring interface, Proc. of 2nd Working Conference on Asynchronous Design Methodologies, London, May 1995, IEEE Comp. Society Press, N.Y., 1995, pp. 32-41.

(3) Synthesis Methods and Tools for Asynchronous Circuits:

8. A.V. Yakovlev. Synthesis of hazard-free asynchronous circuits from generalised signal-transition graphs, Proc. 6th Int. Conf. VLSI Design (VLSI DESIGN'93), Bombay, India, January 1993, IEEE Comp. Society Press, N.Y., 1993, pp. 21-24.
9. A. Yakovlev, A. Petrov [C,E], and L. Rosenblum [C,E]. Synthesis of asynchronous control circuits from symbolic signal transition graphs, Asynchronous Design Methodologies (Ed. S. Furber, M. Edwards), IFIP Transactions A-28, North-Holland, 1993 (Proc. IFIP 10.5 Working Conf., Manchester, March-April 1993), pp. 71-85.
10. A. Semenov, A. Yakovlev, E. Pastor, M. Pena, J. Cortadella, and L. Lavagno Partial order approach to synthesis of speed-independent circuits. Proc. 3rd Int. Symp. on Advanced Research in Asynchronous Circuits and Systems, Eindhoven, Holland, April 1997, IEEE Comp. Soc. Press, pp. 254-265.
11. J. Cortadella [E], M. Kishinevsky [E], A. Kondratyev [E], L. Lavagno [E] and A. Yakovlev. Logic Synthesis of Asynchronous Controllers and Interfaces, Springer Series in Advanced Microelectronics, vol. 8, Springer, 2002, ISBN-3-540-43152-7.
12. D. Shang [P], F.Burns[P], A.Koelmans [C], A.Yakovlev and F. Xia [P]. Asynchronous system synthesis based on direct mapping using VHDL and Petri nets, IEE Proceedings, Computers and Digital Techniques (CDT), Vol. 151, No.3, May 2004, pp. 209-220 (awarded the IEE CDT Premium award for the Best 2004 paper in IEE Proc CDT).

(4) Analysis and Verification of Asynchronous Systems:

13. L.Ya. Rosenblum [E] and A.V. Yakovlev, Analysing semantics of concurrent hardware specifications. Proc. Int. Conf. on Parallel Processing (ICPP89), Pennstate University Press, University Park, PA, July 1989, pp. 211-218, Vol.3.
14. W. Vogler [E], A. Semenov [P] and A. Yakovlev. Unfolding and Finite Prefix for Nets with Read Arcs. Proceedings of CONCUR'98, Nice, France, Sept. 1998, LNCS No. 1466, pp. 501-516.
15. F. Burns [P], A. Koelmans [C] and A. Yakovlev. WCET Analysis of Superscalar Processors Using Simulation With Coloured Petri Nets, Real-Time Systems, The International Journal of Time-Critical Computing Systems, Volume 18, Issue 2/3, May 2000, Kluwer Academic Publishers, pp. 275-288.

(5) Asynchronous Communication Mechanisms:

16. F. Xia [P], A.V. Yakovlev, I.G. Clark [P] and D. Shang [P]. Data communication in systems with heterogeneous timing, IEEE Micro, vol. 22, No. 6, pp. 58-69, Nov./Dec. 2002.

17. A. Yakovlev and F. Xia [P]. Towards synthesis of asynchronous communication algorithms, Int. Workshop on Synthesis of Concurrent Systems, ICATPN'01 and ICACSD'01, Newcastle upon Tyne, June 2001. (Published in: Synthesis and Control of Discrete Event Systems (Eds. B. Caillaud, P. Darondeau, L. Lavagno and X. Xie), Kluwer Academic Publishers, ISBN-0-7923-7639-0, January 2002, pp. 57-75.)

(6) Circuits with metastable behaviour: Arbiters, Synchronisers, A-D converters and Time measurement

18. A. Yakovlev, A. Petrov [E], and L. Lavagno [E]. A Low Latency Asynchronous Arbitration Circuit, IEEE Trans. on VLSI Systems, vol. 2, No. 3, Sept. 1994, pp. 372-377.
19. A. Bystrov, D. Kinniment, and A. Yakovlev. Priority Arbiters. Proc. Sixth Int. Symp. on Advanced Research in Asynchronous Circuits and Systems (ASYNC2000), April 2000, Eilat, Israel, IEEE Computer Society Press, pp. 128-137.
20. D.J. Kinniment [C], O.V. Maevsky [E], A. Bystrov [C], G. Russell [C], A.V. Yakovlev. On-chip structures for timing measurement and test, Microprocessors and Microsystems, Vol. 27, No. 9 (October 2003), pp. 473-483.

(7) Fault-tolerance, Testing, Security:

21. A.V. Yakovlev. Structural technique for fault-masking in asynchronous interfaces. IEE Proceedings E (Computers and Digital Techniques), Vol. 140, No.2, March 1993, pp. 81-91.
22. D. Sokolov [P], J. Murphy [P], A. Bystrov [C] and A.Yakovlev, Design and Analysis of Dual-Rail Circuits for Security Applications, IEEE Transactions on Computers, Vol. 54, No.4, pp. 449-460, April 2005.

Group II (Supporting contributions):

(1) Models of Asynchronous Systems based on Petri nets:

23. A.Yu. Kondratyev [C], L.Ya. Rosenblum [C], and A.V.Yakovlev, Signal graphs: a model for designing concurrent logic. Proc. Int. Conf. on Parallel Processing (ICPP), Pennstate University Press, University park, PA, July 1988, pp. 51-54, Vol.1.
24. A.V. Yakovlev. On limitations and extensions of signal transition graph model for designing asynchronous control circuits. Proc. Int. Conf. on Computer Design (ICCD'92), Cambridge, MA, October 1992, IEEE Comp. Society Press, N.Y., pp. 396-400.
25. J. Cortadella, M. Kishinevsky, L. Lavagno and A. Yakovlev, Deriving Petri Nets from Finite Transition Systems, IEEE Transactions on Computers, Vol. 47, Number 8, pages 859-882, Aug. 1998.

26. A. Yakovlev. Is the Die Cast for the Token Game? (invited paper) Proc. of 23rd International Conference on Applications and Theory of Petri nets (ICATPN 2002), Adelaide, Australia, June 2002, LNCS 2360, Springer, pp. 70-79.
27. A. Bystrov [P], D. Sokolov [P], and A. Yakovlev. Low latency control structures with slack, Proc. of Int. Symp on Advanced Research in Asynchronous Systems and Circuits (ASYNC'03), May 2003, Vancouver, IEEE CS Press, pp. 164-173.

(2) Asynchronous Circuit Design (using Petri net based models):

28. V.I. Varshavsky [C], V.B. Marakhovsky [C], L.Ya. Rosenblum [C], and A.V. Yakovlev. Principles of self-timing and interface models in VLSI systems, Automatic Control and Computer Sciences (translated and published by Allerton Press, USA), vol. 19, No.3, pp.72-78 (1985).
29. V.I. Varshavsky [C], V.B. Marakhovsky [C], L.Ya. Rosenblum [C], and A.V. Yakovlev. Implementation and analysis of the TRIMOSBUS self-clocking interface, Automatic Control and Computer Sciences (translated and published by Allerton Press, USA), vol.19, No.4, pp.80-87 (1985).
30. A. Yakovlev. Designing Self-Timed Systems, VLSI SYSTEMS DESIGN, Vol. VI, No. 9, September 1985, pp. 70-90.
31. A. Yakovlev and A. Petrov [C], Petri nets and asynchronous bus controller design, Revised version (January 1991) of our paper "Petri nets and Parallel Bus Controller design", published at Int. Conference on Applications and Theory of Petri nets, Paris, June 1990, pp. 244-264.
32. A. Semenov [P], A.M. Koelmans [C], L. Lloyd [P] and A. Yakovlev. Designing an asynchronous processor using Petri nets, IEEE Micro, Vol. 17, No. 2 (March/April 1997), pp. 54-64.
33. A. Yakovlev, A.M. Koelmans [C], A. Semenov [P] and D.J.Kinniment [C], Modelling, Analysis and Synthesis of Asynchronous Control Circuits Using Petri Nets, INTEGRATION: the VLSI Journal, Vol. 21 (1996), pp. 143-170.
34. A. Yakovlev, A. Koelmans [C], and L. Lavagno [E]. High level modelling and design of asynchronous interface logic, IEEE Design and Test of Computers, Spring 1995, pp. 32-40.
35. L. Lloyd [P], K. Heron [C], A. Yakovlev, and A.M. Koelmans [C]. Asynchronous microprocessors: from high level model to FPGA implementation. Journal of Systems Architecture vol. 45 (1999), pp. 975-1000, Elsevier.
36. N. Starodoubtsev [E], A. Bystrov [P], and A. Yakovlev. Semi-modular latch chains for asynchronous circuit design. Proc. 10th Int. Workshop on Power and Timing Modelling, Optimization and Simulation (PATMOS'2000), Goetingen, Germany, Sept. 2000, D. Soudris, P. Pirsch, E. Barke (Eds), LNCS 1918, Springer, pp. 168-177.

(3) Synthesis Methods and Tools for Asynchronous Circuits:

37. J. Cortadella[E], M. Kishinevsky[E], A. Kondratyev[E], L. Lavagno[E] and A. Yakovlev, Petrify: a tool for manipulating concurrent specifications and synthesis of asynchronous controllers, IEICE Trans. Inf. & Syst., Vol. E80-D, No.3, March 1997, pp. 315-325.

38. A. Kondratyev[E], J. Cortadella[E], M. Kishinevsky[E], L. Lavagno[E], and A. Yakovlev. Logic decomposition of speed-independent circuits (invited and refereed paper), Proceedings of IEEE, vol. 87, no.2, pp. 347-362, Feb. 1999.
39. A. Kondratyev [E], M. Kishinevsky [E] and A. Yakovlev. Hazard-free implementation of speed-independent circuits, IEEE Trans. on CAD, vol. 17, no. 9, pp. 749-771, Sept. 1998.
40. J. Cortadella[E], M. Kishinevsky[E], A. Kondratyev[E], L. Lavagno[E] and A. Yakovlev, A region-based theory for state assignment in speed-independent circuits, IEEE Trans. on CAD, vol. 16, no. 8, August 1997, pp. 793-812.
41. J. Cortadella[E], M. Kishinevsky[E], A. Kondratyev[E], L. Lavagno, E. Pastor[E], and A. Yakovlev. Decomposition and technology mapping of speed-independent circuits using boolean relations. IEEE Trans. of CAD, Vol. 18, No. 9, Sep. 1999, pp. 1221-1236.
42. J. Cortadella[E], M.Kishinevsky[E], S.M. Burns[E], K.S. Stevens[E], A. Kondratyev[E], L. Lavagno[E], A. Taubin[E] and A. Yakovlev, Lazy Transition Systems and Asynchronous Circuit Synthesis with Relative Timing Assumptions. IEEE Trans. of CAD, Vol. 21, No. 2, Feb. 2002, pages 109-130.
43. J. Carmona [E], J.Cortadella [E], V. Khomenko [C] and A.Yakovlev. Synthesis of Asynchronous Hardware from Petri Nets, Lectures on Concurrency and Petri Nets: Advances in Petri Nets, Lecture Notes in Computer Science, Volume 3098 / 2004, Publisher: Springer-Verlag Heidelberg, ISSN: 0302-9743, ISBN: 3-540-22261-8, pp 345-401.
44. V. Khomenko [C], M. Koutny [C], and A. Yakovlev: Detecting State Coding Conflicts in STG Unfoldings Using SAT. Special Issue on Best Papers from ICACSD'2003, IOS Press, Fundamenta Informaticae 62(2) (2004) 1-21.
45. A. Madalinski [P], A. Bystrov [P], V. Khomenko [C] and A. Yakovlev. Visualisation and resolution of encoding conflicts in asynchronous circuit design, IEE Proc. CDT, vol. 150, No.5, Sept. 2003, pp. 285-293 (Special issue of Best papers at DATE'2003).
46. D. Sokolov [P] and A. Yakovlev, Clockless circuits and system synthesis, IEE Proceedings, Computers and Digital Techniques, vol.152, No.3, May 2005, pp. 298-316.
47. N. Starodoubtsev [E], S. Bystrov [E], and A. Yakovlev. Monotonic circuits with complete acknowledgement, Proc. of ASYNC'03, Vancouver, IEEE CS Press, pp. 98-108.
48. A. Bystrov [P] and A. Yakovlev. Asynchronous Circuit Synthesis by Direct Mapping: Interfacing to Environment, Proc ASYNC'02, Manchester, April 2002, IEEE CS Press, 127-136.
49. D. Sokolov [P], A. Bystrov [P] and A. Yakovlev. STG optimisation in the direct mapping of asynchronous circuits, Proc. DATE 2003, Munich, March 2003, pp. 932-937.
50. D.Shang [P], F.Xia [P] and A.Yakovlev, Asynchronous Circuit Synthesis via Direct Translation, Proc. ISCAS'02, Scottsdale, Arizona, May 2002, IEEE, Volume III, pp. 369-372.

(4) Analysis and Verification of Asynchronous Systems:

51. A. Semenov [P] and A. Yakovlev. Combining partial orders and symbolic traversal for efficient verification of asynchronous circuits. Proc. IFIP Int. Conference on Computer Hardware Description Languages, (CHDL'95),Chiba, Japan, August-September 1995, pp. 567-573.
52. F.B. Burns, A.M. Koelmans, A.V. Yakovlev. Analysing superscalar processor architectures with coloured Petri nets. Int. Journal on Software Tools for Technology Transfer, Vol.2, No.2, December 1998, Springer, pp. 182-191.
53. A.Burns [E], A.J.Wellings [E], A.M.Koelmans [C], M.Koutny [C], A.Romanovsky [C], A.Yakovlev. On Developing and Verifying Design Abstractions for Reliable Concurrent Programming in Ada. Ada Letters, v. XXI, no. 1, March 2001, pp. 48-55.
54. I. Mitrani [C], A. Yakovlev. Tree Arbiter with Nearest-Neighbour Scheduling. Proc. of the 13th International Symposium on Computer and Information Sciences (ISCIS'98), 26-28 October, Belek-Anatlya, Turkey. In: Advances in Computer and Information Sciences'98 (Eds. U. Gudukbay, T. Dayar, A. Gursoy, E. Gelenbe) Concurrent Systems Engineering Series Vol. 53, pp. 83-92, ISBN 90-5199-405-2 (IOS Press*).*

(5) Asynchronous Communication Mechanisms:

55. F. Xia [P], A. Yakovlev, D. Shang [P], A. Bystrov [P], A. Koelmans [C], and D. Kinniment [C]. Asynchronous Communication Mechanisms using Self-timed Circuits. Proc. Sixth Int. Symp. on Advanced Research in Asynchronous Circuits and Systems (ASYNC2000), April 2000, Eilat, Israel, IEEE Computer Society Press, pp. 150-159.
56. F. Xia [P], F. Hao [P], I. Clark [P], A.Yakovlev and E.G. Chester [C], Buffered asynchronous communication mechanisms, Proc. Fourth Int. Conf. Aplications of Concurrency to System Design (ACSD 2004), Edited by M. Kishinevsky and Ph. Darondeau, Hamilton, Ontario, Canada, 16-18 June 2004, IEEE Press, pp. 36-45.
57. F. Hao [P], F. Xia [P], E.G. Chester [C], A. Yakovlev and I. G. Clark [P]. MATLAB Models of ACMs in Control Systems, 1st International Conference on Informatics in Control, Automation and Robotics (ICINCO-2004), Setubal, Portugal, 25-28 August 2004, INSTICC Press, vol.3, pp. 54-61.
58. A. Yakovlev, F. Xia [P] and D. Shang [P]. Synthesis and implementation of a signal-type asynchronous data communication mechanism. In Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC'01), Salt Lake City, pages 127-136. IEEE Computer Society Press, March 2001.

(6) Circuits with metastable behaviour: Arbiters, Synchronisers, A-D converters and Time measurement

59. J. Cortadella [E], L. Lavagno [E], P. Vanbekbergen [E], and A. Yakovlev. Designing Asynchronous Circuits from Behavioural Specifications with Internal Conflicts, Proc. Int. Symp. on Advanced Research in Asynchronous Circuits and

Systems (ASYNC'94), Salt Lake City, Nov. 1994, IEEE Comp. Society Press, N.Y., pp. 106-115.
60. A. Bystrov [P] and A. Yakovlev. Ordered arbiters. Electronics Letters, 27th May 1999, Vol. 35, No. 11, pp. 877-879.
61. I.G. Clark [P], F. Xia [P], A.V. Yakovlev and A.C. Davies [E]. Petri net models of latch metastability, Electronics Letters, 2nd April 1998, Vol. 34, No.7, pp. 635-636.
62. D.J. Kinniment [C], B. Gao [P], A. Yakovlev and F. Xia [P]. Towards asynchronous A-D conversion, Proc. 4th Int. Symp. on Advanced Research in Asynchronous Circuits and Systems, March-April 1998, San Diego, CA, IEEE Computer Society Press, pp. 206-215.
63. D.J. Kinniment [C] and A.V. Yakovlev. Low power, low noise micropipelined flash A-D converter, IEE Proc. Circuits, Devices Systems, vol. 146, no.5, October 1999, pp 263-267.
64. D. J. Kinniment [C], A.V. Yakovlev, and B. Gao [P]. Synchronous and Asynchronous A-D Conversion, IEEE Transactions on VLSI systems, Vol. 8 No. 2 Apr. 2000, pp. 217-219.
65. D.J.Kinniment [P], A. Bystrov [P], A.V. Yakovlev. Synchronization Circuit Performance, IEEE Journal of Solid-State Circuits, vol. 37, no. 2, Feb. 2002, pp. 202-209.
66. D.J. Kinniment [C], O. Maevsky [E], A. Bystrov [P], G. Russell [C] and A. Yakovlev, On-chip Structures for Timing Measurements and Test, Proc ASYNC'02, Manchester, April 2002, IEEE CS Press, pp. 190-197.
67. A.M. Abas [P], A.Bystrov [P], D.J. Kinniment [C], O.V. Maevsky [E], G. Russell [C], and A.V.Yakovlev. Time difference amplifier, Electronics Letters, vol. 38, no. 23, pp. 1437-1438, 7 Nov. 2002.
68. D.J.Kinniment [C] and A.V.Yakovlev. Low latency synchronisation through speculation, PATMOS 2004, Santorini, September 2004, LNCS 3254, Springer, pp. 278-288.
69. O. Maevsky [E], D.J. Kinniment [C], A.Yakovlev, A. Bystrov [P]. Analysis of the oscillation problem in tri-flops, Proc. ISCAS'02, Scottsdale, Arizona, May 2002, IEEE, volume I, pp.381-384.

(7) Fault-tolerance, Testing, Security:

70. V.I. Varshavsky [C], V.Ya. Volodarskii [E], V.B. Marakhovsky [C], L.Ya. Rosenblum [C], Yu.S. Tatarinov [P], and A.V. Yakovlev. Hardware implementation of protocols for a fault-tolerant self-synchronous ring channel, Automatic Control and Computer Science (translated from Russian, Allerton Press, USA), vol.22, No.6, pp. 59-67 (1988).
71. V.I. Varshavsky [C], V.Ya. Volodarskii [E], V.B. Marakhovsky [C], L.Ya. Rosenblum [C], Yu.S. Tatarinov [P], and A.V. Yakovlev. Algorithmic and structural organisation of test and recovery facilities in a self-synchronous ring, ibid., vol.23, No.1, pp.53-58 (1989).
72. V. Varshavsky [E], V. Marakhovsky [E] and A. Yakovlev. Towards Self-Checking and Self-Recovery in Self-Timed Embedded Systems. Proc. IEEE International Workshop on Embedded Fault-Tolerant Systems, Dallas, Texas, September 1996.

73. D. Shang [P], F. Xia [P] and A. Yakovlev. Testing a Self-Timed Asynchronous Communication Mechanism (ACM) VLSI Chip, Proc. IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, Gyor, Hungary, 18-20 April, 2001, pp. 53-56.
74. D. Shang [P], F. Burns [P], A. Bystrov [C], A. Koelmans [C], D.Sokolov [P] and A.Yakovlev. A low and balanced power implementation of the AES security mechanisms using self-timed circuits, PATMOS 2004, Santorini, September 2004, LNCS 3254, pp. 471-480.
75. D. Sokolov [P], J.Murphy [P], A.Bystrov [C] and A. Yakovlev. Improving the security of dual-rail circuits, Proc. CHES 2004, M. Joye and J.-J. Quisquarter (Eds), August 2004, LNCS 3156, Springer, pp. 282-297.
76. J. Murphy [P], A. Bystrov [C]and A.Yakovlev, Self-Checking Circuits for Security Applications, 11th Annual International Mixed-Signals Testing Workshop (IMSTW'05), Cannes, France, June 2005, pp. 278-285.
77. D. Shang [P], A. Bystrov [C], A. Yakovlev and D. Koppad [P], On-line Testing of Globally Asynchronous Circuits, Proc. 11th International Online Testing Symposium (IOLTS'05), St. Rafael, France, July 2005, IEEE CS Press, pp. 135-140.