Behavioural Refinements for Asynchronous Circuit Synthesis
(Final Report on EPSRC Research Grant, GR/M94359)

A.V.Yakovlev*

1 Background/Context

Research in asynchronous circuit design, both in the UK and abroad, has shown an upsurge of interest in the last decade. A number of symposia on advanced research in asynchronous systems have been held in USA, Japan and Europe; they have become an annual event and attract an ever increasing industrial sponsorship and participation. Promoting asynchronous design methods into practice has been most remarkable in Manchester, where the Amulet group, led by S. Furber, has developed a family of ARM-compatible industrial strength microprocessors, and a number of embedded applications. The group's most recent developments have also been in the area of design tools. The role of and demand for a flexible design flow is rapidly increasing as was emphasized at the last two (EU-funded) ACiD-WG workshops. Examples of continued industrial success are Philips and Theseus logic, where design flow development was given top priority at the start of all developments, confirm the argument. One of the key stages in a successful design flow, with a good balance between productivity and optimality, would be automatic synthesis from behavioural descriptions.

The BREACH project has been part of Newcastle's steady involvement in the research in asynchronous synthesis over the last ten years. During the 90s this research found its support from EPSRC through the ASAP (GR/J52327) grant, in collaboration with the University of Britsol (Prof. E. Dagless), and visiting fellowships (GR/J72486, J78334, L24038) of Professors M. Kishinevsky, L. Lavagno and A. Kondratyev, which was accompanied by travel grants from the British Council and Spanish Ministry of Science and Culture to support collaboration with the Group of Prof. J. Cortadella from UPC (Barcelona). The main result of this work was a set of synthesis methods and a software tool Petrify described in the forthcoming monograph [7].

The research carried out under the BREACH (Visiting Fellowship for Prof. N. Starodoubtsev) grant was a natural progression from those developments. This research tackled a number of limitations of the state-of-the-art of Petrify (and more indirectly, other relevant tools such as e.g. MINIMALIST) and was aimed at extending the power of decomposition techniques and the applicability of synthesis tools for a range of design technologies – a critical issue for the process of promulgation of asynchronous design techniques into a broader spectrum of system-on-a-chip applications. Particular emphasis has been on improving decomposition of asynchronous control circuits using behavioural model refinement (as opposed to the more traditional approach of functional logic decomposition) within the Petri net based synthesis environment.

This research was closely related with other projects being conducted at Newcastle, such as the ones on model visualisation for asynchronous design (MOVIE, GR/M94366), on asynchronous communication mechanisms (COMFORT, GR/L93775), and most recently, the BESST project (GR/R16754) focused on behavioural synthesis of systems with heterogeneous timing.

The results of BREACH have been reported in six academic publications of Prof. Starodoubtsev together with our group at Newcastle and with his research team in St.Petersburg, including conference proceedings and presentations at seminars and workshops. In particular, these results were presented to the international microelectronic design community (PATMOS workshops) and asynchronous circuit design community, internationally (ACiD-WG, AINT workshops) and within

*Department of Computing Science, University of Newcastle upon Tyne, Newcastle upon Tyne NE1 7RU; 0191-222-8184; alex.yakovlev@ncl.ac.uk

the UK (Asynchronous UK Forum). Through these presentations the project contributed to wider understanding of the relationship between Signal Transition Graphs and asynchronous circuits, decomposition and synthesis in standard cell libraries among such industrial companies as IBM, Intel, Philips, AT Microelectronics, Theseus Logic, CSEM. BREACH has impact on the writing of the monograph [7] and other publications and presentations.

2 Key Advances and Supporting Methodology

The original objectives of the project were specified in a number of tasks. These are repeated here along with the contributions made to each one.

2.1 Methods for behavioural refinement of asynchronous control circuits

Existing synthesis tools that work with STGs and state graphs perform decomposition and technology mapping of complex gates using the traditional approach, which is based on boolean function decomposition. In Petrify, for example, boolean factorisation acts as a generator of possible candidates for circuit decomposition. In order to satisfy the speed-independence of the new circuit Petrify must guarantee the property of acknowledgement for the signals created by such a decomposition. In most cases this property is not automatically fulfilled, which results in Petrify having to search, through the reachable state space, for possible places where to insert additional acknowledging connections. This often leads to sub-optimal solutions because the tool avoids explicit introduction of internal signals in the model at the behavioural level. Moreover, in mapping boolean functions to logic gates, Petrify does not aim at eliminating 'bubbles' (input invertors) in the implementation functions. This may in practice be plausible due to the inverter delay being realistically smaller than the delay on a path of gates, but as technology migrates and parameters, such as threshold, deviate more such an assumption may become overly optimistic. The creation of new bubbles is concerned with the fact that new signals are inserted on one-by-one (cf. single-rail) basis.

In order to avoid problems with input inverters conditions for STG normalcy, and hence monotonicty of the logic implementation, have been formulated at the STG level in [5]. They use the idea of a partial state associated with a subset of Petri net places (e.g, pre-places of a transition). This approach also enables to handle the fanins of signals using their trigger signals. A method based on the refinement of an STG specification in order to guarantee the satisfaction of complete state coding and normalcy, automatically satisfying the speed-independence and behavioural equivalence has been presented in [5, 3]. A particular elegance of this refinement is a heuristic that is called a flip-flop insertion. This is an insertion of complementary signals in the STG, which implicitly adds to the circuit a pair of negative gates, in many cases realising a flip-flop.

Most recent investigation of STG refinement at the PN level uses the idea of separation of concurrency and choice [8]. It is related to our work, under the MOVIE project, on Petri net unfoldings and their application to the visualisation of STGs, analysis and resolution of CSC conflicts and interactive synthesis of high-performance asynchronous controllers [10, 9]. Studying this refinement deeper is the subject of future work.

2.2 Speed-independence of modern self-timed implementation technologies

Speed-independent circuits, in their classical definition of D.E. Muller and W.S. Bartky are insensitive to gate delay variations. They are characterised by the property of acknowledgement with respect to the outputs of their gates. In order to 'gracefully ignore' the effect of wire delays, the idea of isochronic forks and the notion of quasi-delay-insensitive circuits is often applied at present. The use of deep-submicron technology will increase the role of wire delays, compared to gate delays, as well as the effect of variations in other parameters such as switching thresholds of transistors. It is much more difficult to guarantee acknowledgement at the level of inputs to the gates, effectively allowing ends of forks to introduce arbitrary delays.

We have introduced and performed a preliminary analysis of circuits that are free from isochronic forks [4]. The elimination of forks between gates is achieved by using a special type of gates. For example, a gate whose main functional output is NAND will also have an additional output that is an inversion of one of the inputs. In standard SI circuits, the problem with isochronic forks is effectively that of acknowledging inputs that are context (not trigger) to the gates. Such inputs, for particular transitions, are left dangling (cf. orphans in Theseus' NCL). The use of our special gates allows the acknowledgment of such dangling signals using the additional, inverting, outputs. Our report in [4] presented a number of examples of realistic control circuits, synthesised using our STG-level refinement ideas, which has opened up a number of new interesting problems for further research into refinement techniques. Some canonical approaches, like those based on direct mapping of Petri nets into netlist of cells, implemented with isochronic fork free fragments, might be pursued. However, the issue of performance penalty compared to less robust speed-independent solutions could be crucial. Analysis and experimental results showed certain difficulties in implementing such circuits at the low level for the widest possible class of specifications and with full insensitivity to isochronic forks. It seems that the pragmatic approach should be here like in testing (where a certain percentage of testability is guaranteed) to construct circuits with a certain, practically acceptable, percentage of isochronic forks eliminated.

We have explored the conditions for semi-modular (hence hazard-free) operation for complex gate implementations in restricted gate array ASIC libraries such as IBM SA-12E (see next section) [1].

2.3 Development of algorithms for decomposition and technology mapping of asynchronous control circuits

Algorithms and software for the analysis of normalcy and CSC conditions avoiding the full state space traversal of the STG were presented in [2, 6]. These algorithms are now part of TaxoSynthesis (the tool developed by Starodoubtsev's team).

A semi-automatic design flow, combining TaxoSynthesis, Petrify and VHDL simulation tools has been significantly advanced in [2] and [3]. The ideas for such a tool originate in [14, 15]. The results of experiments with this flow, producing circuits built of monotonic gates only, and the STG benchmarks showed that the gain in robustness of the implementation (no input inverters) did not lead to loss in area or power or significant speed penalty (intuitively one would expect some loss in speed due to providing complete acknowledgement of all signals in the circuit). Quite opposite, there was an average saving of 28% in area and 23% in power consumption, with only an average 6% loss in circuit speed. This was due to a much better correspondence between the structure and the behaviour in the monotonic circuits obtained through STG-level behavioural refinement.

In the course of the project, we came across an efficient structural refinement technique called semi-modular latch chaining. Quite interesting on its own, it can be characterised as a technology mapping method specific to asynchronous circuits to its inherent hazard-freedom. This technique allows decomposing complex asymmetric C-gates (possibly obtained from Petrify's synthesis) into canonical chains of set-dominant and reset-dominant latches with interlacing feedbacks. Those latches can be easily mapped into a restricted gate array ASIC library, such as IBM SA-12E, which includes AO12, AOI12, OA12, OAI12. The technique was described in [1] and illustrated by entirely new implementations of various useful circuits such as a three input C-element, a toggle and an edge-triggered pipeline stage controller. The idea of latch chaining was also investigated at the behavioural (STG) level and later incorporated into the above-mentioned design flow [3].

3 Project Plan Review

There have been no major deviation from the original plan of investigation. In terms of research visits undertaken by Prof. Starodoubtsev, it was possible to arrange three visits instead of planned two, which was partly concerned with Prof. Starodoubtsev's move to Japan in the beginning of 2001.

4 Research Impact and Benefits to Society

Our research group at Newcastle considers this research to have been very successful. All of the tasks have been completed and in many cases the results exceeded original expectations considerably. For example, those on latch chaining were a complete surprise to us. Prof. Starodoubtsev's visits have been very beneficial to the Newcastle's research in asynchronous design as a whole. Prof. Starodoubtsev took part in our weekly seminar of the Asynchronous Systems laboratory [11] and had interactions with RAs and PhDs at Newcastle and in the UK (his talk at the 9th UK Asynchronous Forum in Cambridge). One of the results was the impact of those interactions on our analysis of CSC and normalcy presented in an important DATE'02 publication [9]. Another strong impact of this visiting fellowship was the continued collaboration between our Newcastle team and Prof. Starodoubtsev's group. For example, one of the members of his group Alexander Smirnov, who has been recently working at the Polytechnic University of Catalonia with Prof. Jordi Cortadella (the principal author of the Petrify tool), was able to contribute significantly to the development of the idea of combined (with Petrify) interactive synthesis flow for monotonic gate implementation [2, 3]. The recent visit of A. Smirnov to Newcastle [11, 8] and discussions about Petri net transformations have been beneficial for Newcastle's research [10].

The overall list of publications and presentations related to this grant is a tangible measure of the output from BREACH.

5 Explanation of Expenditure

The expenditure plans in the original research proposal have been followed without changes. Funds provided by EPSRC were spent solely on Prof. Starodoubtsev's travel, accommodation and subsistence.

6 Further Research/Dissemination Activities

The research outcome from this grant is directly applicable in the fields of asynchronous circuit design and modelling and analysis of concurrent systems. It is of direct benefit to our three EPSRC grants, one on model visualisation for asynchronous design (MOVIE, GR/M94366), on asynchronous communication mechanisms for SOCs held jointly between Newcastle and Kingston Universities (COHERENT, GR/R32666/R32895), and the BESST project (GR/R16754) focused on behavioural synthesis of systems with heterogeneous timing. It will also be useful to the UK research in asynchronous design, supported by EPSRC at Manchester and Cambridge (projects on Balsa synthesis environment and GALS). The refinement-based synthesis of STG and analysis of related properties is now part of PhD research of V. Khomenko and A. Madalinski, D. Shang.

The ideas of model refinement at the behavioural (PN and STG) level are of long-term value. While currently related to the matters concerned with the CMOS technology (e.g., STG normalcy and monotonic gate implementation, minimisation of context dependency and isochronic fork elimination), they are part of a fundamental relationship between the dynamic causal and concurrency structure present in specification models and the physical interconnection structure of circuit implementation. Studying such relationships is crucial for further progress in designing systems for new technologies.

This research has had its impact on preparing two advanced tutorials at the international conferences: Adv. Tutorial on Hardware Design and Petri Nets at ICATPN2000 in Aarhus [12], June 2000, and Tutorial on Asynchronous Circuit Design in ASP-DAC-VLSI, Bangalore, Jan. 2002 [13]. It also affected preparing the monograph on STG-based synthesis [7], which will disseminate this knowledge to the wider public through special courses on Asynchronous Design (ACiD-WG Summer School in Grenoble in July 2002). It will also be put forward in our invited talk and paper at the next ICATPN in Adelaide 2002 and in preparing an edited monograph "Advances in Concurrency and Hardware Design" to be published by Springer in 2002.

References

[1] N. Starodoubtsev, A. Bystrov, and A. Yakovlev. Semi-modular latch chains for asynchronous circuit design. *Proc. Power and Timing Modeling, Optimization and Simulation (PATMOS)*, volume 1918 of *Lecture Notes in Computer Science*, pages 168-177, September 2000.

[2] A. Smirnov, M. Goncharov, I. Klotchkov, and N. Starodoubtsev. A Technique to Automate STG Analysis and Refinement for CSC and Normalcy, *Handouts of the First ACiD-WG Workshop (Framework 5)*, Neuchatel, Switzwerland, 12-13 February, 2001.

[3] N. Starodoubtsev, S. Bystrov, M. Goncharov, I. Klotchkov and A. Smirnov. Towards Synthesis of Monotonic Asynchronous Circuits from Signal Transition Graphs. *Proceedings of the Second International Conference on Application of Concurrency to System Design (ACSD 2001)*, June 2001, Newcastle upon Tyne, IEEE Comp. Society, 2001, pp. 179-188.

[4] N. Starodoubtsev and A. Yakovlev. Isochronic Fork-Free Asynchronous Circuits, *Proc. 9th Asynchornous UK Forum*, Cambridge, Dec. 2000.

[5] N. Starodoubtsev, M. Goncharov, I. Klotchkov and A. Smirnov. Synthesis of asynchronous interface circuits by STG refinement. In: A. Yakovlev and R. Nouta (Eds.) *Proceedings of Int. Workshop on Asynchronous Interfaces: Tools, techniques, and Implementations (AINT'2000)*, TU Delft, The Netherlands, July 2000, ISBN 90-5326-037-4, pp. 65-74.

[6] A.B. Smirnov, N. Starodoubtsev, I.V. Klotchkov and M. Goncharov, A technique to automate STG analysis and refinement for CSC and normalcy, *Proc. Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Yverdon-Les-Bains, Switzerland ,September 2001.

[7] J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev, *Logic Synthesis of Asynchronous Controllers and Interfaces*. Springer Series in Advanced Microelectronics, vol. 8, Springer, 2002, ISBN-3-540-43152-7.

[8] A.B. Smirnov, Clean Choice-Concurrency Petri Nets *Proc. 11th Asynchornous UK Forum*, Cambridge, Dec. 2001.

[9] V. Khomenko, M. Koutny and A. Yakovlev. Detecting state coding conflicts in STGs using integer programming, CS-TR-736 Department of Computing Science, University of Newcastle, 2001 (to appear in Proc. DATE'02, Paris, March 2002).

[10] A. Bystrov, M. Koutny and A. Yakovlev, Visualization of partial order models in VLSI design flow, CS-TR-744, Department of Computing Science, Univ. of Newcastle upon Tyne, 2001 (to appear in Proc. DATE'02, Paris, March 2002)

[11] Asynchronous Systems Laboratory Seminars: http://www.cs.ncl.ac.uk/events/ASL/

[12] J. Cortadella, L. Lavagno and A. Yakovlev, Advanced Tutorial on Hardware Design and Petri Nets; http://www.cs.ncl.ac.uk/people/alex.yakovlev/home.formal/talks/hwpn-adv-tut/

[13] J. Cortadella, J.Garside and A.Yakovlev, Tutorial on Logic Design of Asynchronous Circuits, ASP-DAC/VLSI Design 2002;
http://www.cs.ncl.ac.uk/people/alex.yakovlev/home.formal/talks/vlsi2002/vlsi02.zip

[14] N.Starodoubtsev, A. Yakovlev and S. Petrov, Use of VHDL-based environment for interactive synthesis of asynchronous circuits. *Proc. VHDL Forum in Europe Spring'96 Working Conference*, Dresden, Germany, May 1996, Shaker Verlag, Aachen, pp. 21-33.

[15] M.Goncharov, I. Klotchkov, A. Smirnov and N. Starodoubtsev. Timing extension of STG model and a method to simulate timed STG behaviour in VHDL environment. *Proceedings of the First International Conference on Application of Concurrency to System Design (ACSD 1998)*, March 1998, Fukishima, Japan, IEEE Comp. Society, 1998, pp. 120-129.